# Nokia's System For TC-STAR EPPS English ASR Evaluation Task

**Imre Kiss, Jussi Leppanen, Sunil Sivadas**

Multimedia Technologies Laboratory,
Nokia Research Center,
Tampere, Finland.
{imre.kiss, jussi.leppanen, sunil.sivadas}@nokia.com

**Abstract**

This paper describes Nokia's work on complexity reduction of automatic speech recognition (ASR) algorithms. The aim of this paper is to explore algorithms for reduction of memory foot-print of acoustic models. In this paper we compare the performance of speech recognition systems based on hidden Markov models (HMM) with quantized parameters (qHMMs) and subspace distribution clustering hidden Markov models (SDCHMMs). The quantized models were tested on 2006 TC-STAR English parliamentary speech transcription task. It is shown that as low as 4-bits can be used for qHMMs and 2 bits/subspace for SDCHMMs without sacrificing recognition performance.

## 1. Introduction

Spoken language translation (SLT) has a lot of potential applications in mobile devices. To bridge the language barrier between two users, SLT can be used in translating the speech from speaker's language to that of the listener's language. The first step is converting the source speech to text (ASR) and then translating the text in target language (MT) and finally converting the text to speech (TTS). The performance of different components in SLT has improved over the past few years to a point where it can be deployed in domain specific systems in real life.

In the case of resource constrained mobile devices, it is beneficial to have small footprint and low complexity implementation of the algorithms for better user experience. In this paper we focus on evaluating small footprint acoustic modelling techniques for embedded ASR systems. State of the art ASR systems use Continuous Density HMMs (CDHMM) for acoustic modelling. For CDHMM based continuous speech recognisers, the number of densities in the acoustic models is in the order of tens of thousands. If the parameters of the densities are stored in floating point format they occupy a lot of space in memory. Moreover, the largest computationally intensive component in ASR is the calculation of likelihood of the feature vector given a Gaussian density.

In this paper we examine two approaches to reduction in foot-print size of acoustic models and computation without sacrificing the recognition performance. In Sections 2 and 3 we describe the footprint reduction techniques. In Section 4 we compare the properties of these techniques, and continuous density HMMs. Experiments and results are explained in Section 5 and finally conclusions in Section 6.

## 2. Subspace Distribution Clustered HMMs

Subspace distribution clustering HMMs (SDCHMMs) were introduced by Bocchieri and Mak in [1]. They were shown to provide computational and memory savings, without significantly degrading recognition performance, when compared to continuous density HMMs. The basic idea is similar to density tying, but the tying is done on subspace distributions instead of the full-space distributions. The SDCHMM implementation used in this work differs slightly from the one described in [1] and [2]. It is explained briefly below.

First, CDHMMs are trained. The SDCHMMs are obtained from CDHMMs by first dividing the continuous distributions of the original model set into orthogonal subspaces and then clustering these subspace distributions into prototype distributions. The clustering is done separately for every subspace, so that as a result there is a separate prototype set (codebook) for each subspace. The parameters of the original distributions can then be replaced by indices to the codebooks.

The clustering of the continuous distributions is done using a binary, divisive k-means clustering algorithm. The algorithm starts with all distributions belonging to a subspace in a single cluster. After each iteration, each cluster is split into two and then k-means is run for a few iterations. The algorithm is run until the desired number of clusters (codebook size) is obtained. The distance measure used in the k-means algorithm is the Bhattacharya distance [3]. Before clustering can be done, the subspaces have to be defined. This is done by finding the features that correlate the most and grouping them together to form the subspaces. The multiple correlation measure, for example, can be used to find the most correlated features [2]. In this paper, all experiments using SDCHMMs are performed with single-feature subspaces (these were found to work best in our tests). Thus, subspace definition was not needed in our experiments.

.

## 3. Quantized HMM

Quantized HMMs (qHMMs) were proposed in [4]. They reduce memory consumption substantially while maintaining the high recognition accuracy of continuous density HMMs.

Like in the case of SDCHMM, qHMMs are also built from continuous density HMMs. The mean and variance parameters of CDHMMs are scalar quantized. to obtain qHMM. Here only two global quantizers are used, one for the mean values and one for the variance values. The use of one quantizer for all mean values and one for all variance values is potentially problematic as the dynamic range of the feature components varies from component to component. By normalizing the feature space to zero mean and unity variance and assuming that the features are uncorrelated a single quantizer can be used for all the mean components and another one for all the variance components. This is in contrast with the SDCHMM case, where there is a separate quantizer (or codebook) for each subspace and thus no normalization is required. The actual quantization is done using a non-linear Lloyd-Max quantizer. The distance measure used during training of both the mean and variance quantizer was the Euclidian distance.

Figure 1 shows the procedure of deriving qHMM and SDCHMM from a trained CDHMM based acoustic model set.

# 4. Comparison

## 4.1. Codebooks

Figure 2 and 3, below, show the mean vs. inverse standard deviation scatter plots of the 2nd cepstral coefficient with 4-bit qHMM and SDCHMM centroids superimposed on them. The differences in the codebooks for SDCHMMs and qHMMs can be seen quite clearly. The SDCHMM codebook elements are clearly located on the dense regions of the scatter plot. For qHMMs, the placement of the codebook elements does not appear to be very optimal. This is explained by the fact that the same codebook is used for all features and is optimal over all of them.

## 4.2. Memory footprint

When using SDCHMMs or qHMMs, the mean and variance values of the densities are replaced by indices of the codebooks. The space needed to store the indices depends on how large the codebooks are. For example, 39 bytes are needed to store a single density when 8-bit codebooks and 39 element feature vectors are used. This is significantly lower than for the continuous case where (32bits x 2 x 39=) 312 bytes are needed (assuming that 32-bit floating point numbers are used for the mean and variance values). Since the densities take up majority of the space required for the whole model set, significant memory saving can be achieved. Storing the codebooks, however, requires memory, but this is usually insignificant compared to the memory required for the densities. This is the case also for SDCHMMs, where a codebook is required for every subspace.

## 4.3. Probability calculation and feature quantization

During decoding, the calculation of the state probabilities for mixture Gaussians in log-domain is done using the following formula:

$$\log b(x) = \log \sum_{k=1}^{K} \exp\left\{ \log\left( w_k \frac{1}{\prod_{i=1}^{N} \sqrt{2\pi\sigma_{ki}^2}} \right) - \sum_{i=1}^{N} \frac{(x_i - \mu_{ki})^2}{2\sigma_{ki}^2} \right\} \quad (1)$$

where K is the number of densities and N is the feature vector dimension. For each density, there are two parts, a constant and the Mahalanobis distance to the feature vector x.
When using SDCHMMs or qHMMs, once a feature vector is obtained, the Mahalanobis distances can be calculated for the codebook elements. Once this is done, the actual probabilities for the densities can be calculated by summing up the appropriate distances for each density. In the qHMM case, the number of distance calculations is less than in the SDCHMM case because the same codebook is used for all features. For example, when using 39 element feature vectors and 4-bit codebooks, the number of distance calculations that are needed for qHMMs is 16, one for each codebook element. For SDCHMM the corresponding number is (16x39=) 624. This is quite many compared to the qHMM case, but still very efficient (the number of summations needed to calculate the probabilities for 3000 densities for example is still (39 x 3000=) 117000).

The probability calculation can be further speeded up by the use of feature quantization [5]. When the incoming feature vectors are quantized, the Mahalanobis distances of the above equation can be calculated before the decoding begins. Here again the number of distance calculations is greater for SDCHMMs than for qHMMs.

# 5. Experiments

## 5.1. Recognition system and acoustic model sets

As part of the TC-STAR 2006 EPPS English transcription task, we evaluated the acoustic model compression schemes.

The front-end used in the experiments was based on FFT-derived Mel cepstral coefficients and their first and second order derivatives (39 coefficients in total). Recursive mean removal was applied on all components of the resulting feature vectors, and the variance of the energy component and its derivatives was normalized to unity [6].

The baseline acoustic model sets used for the tests contained left to right 3-state cross-word context dependent phoneme models with 16 densities per state. The model sets were trained on an in-house training set containing about 200 hours of speech data from Wall Street Journal (US Engligh), Speecon US and internally recorded databases. The states are clustered using a decision tree. The final model set had about 35,000 Gaussian densities in it.

The recogniser is HTK [7] based. The word lattices were obtained from LIMSI. Lexicon had about 17,000 words in it. Pronunciations for many of the words were generated using an automatic text to phoneme mapping tool. Language model scores were obtained from the lattice. The language model scale factor was tuned on a small development data set.

Using the CDHMMs, a pair of qHMM model sets was trained. One set using 5-bit quantization for the density means and 3-bit quantization for the variances, and the other set using 3 and 1 bits, respectively. The recognition accuracies for these models can be seen in Table 1, labeled as qHMM *ym+zv*, where y the number of bits used in quantizing the means and z the number of bits used for quantizing the variances. The 5+3 and 3+1 quantizations used here were chosen for the reason that they allow convenient packing of the mean-variance pairs into bytes. Similarly, two SDCHMM systems were also trained with 2-bits and 4-bits codebooks. Test consisted of recordings of English portion of European parliamentary speech from plenary sessions. The total duration of test data is about 3 hours. It contains about 40 speakers, out of which 30 are native speakers and 10 are non-native. The recognition rates can be found in Table 1, labeled similarly to the qHMMs.

| Model set | Word Error Rate(%) |
|---|---|
| Baseline (CDHMM) | 18.3 |
| qHMM 5m+3v | 18.3 |
| qHMM 3m+1v | 18.4 |
| SDCHMM 4bits/stream | 18.4 |
| SDCHMM 2bits/stream | 18.6 |

Table 1 Word Error Rate (WER) on parliamentary English speech.

From the above results, it can be seen, that using a 5+3 bit codebook for qHMMs and 4-bit codebooks for SDCHMMs has little effect on the performance. When reducing the codebook size to 2 bits for SDCHMM and 3+1bits for qHMMs, further degradation of the performance can be seen. But the difference is statistically insignificant.

## 5.2. Memory figures

As mentioned before, significant saving in the memory footprint of the acoustic models can be achieved when using qHMMs or

SDCHMMs. In Table 2, below, the memory required to store the densities of the various acoustic model sets used in these experiments is shown. For the baseline models, it is assumed that the mean and variance values are represented using 32 bit floating-point numbers. While the memory needed for the densities is similar for qHMMs and SDCHMMs, the memory needed to store the codebooks is not. Because of the use of separate codebooks for each stream, the codebooks of the SDCHMMs require more memory. However, since the densities require much more memory, the memory needed for the codebooks is not very significant. It can be seen from the table that a reduction of up to 90% in size can be obtained without any degradation in recognition performance.

| Model set | Model size (MB) |
|---|---|
| Baseline (CDHMM) | 11 |
| qHMM 5m+3v | 1.1 |
| qHMM 3m+1v | 0.7 |
| SDCHMM 4bits/stream | 1.9 |
| SDCHMM 2bits/stream | 1.8 |

Table 2 Memory needed for the storage of densities and codebooks of qHMMs and SDCHMMs.

## 6. Conclusions

We presented Nokia's work on small foot-print acoustic modeling algorithms and compared the performance of two small footprint acoustic modeling techniques, qHMMs and SDCHMMs on TC-STAR European Parliamentary speech data. Compared with continuous density HMMs, both techniques provided memory savings and probability calculation speed up without sacrificing recognition accuracy significantly. The memory footprint of qHMMs and SDCHMMs was found to be much smaller than for the continuous density HMMs. The densities of the qHMM and SDCHMM model sets could be represented in only 10% of the bytes needed for the continuous case. Moreover, the memory footprint of qHMMs was slightly smaller than the SDCHMM memory footprint because of the differences in the number of codebooks used.

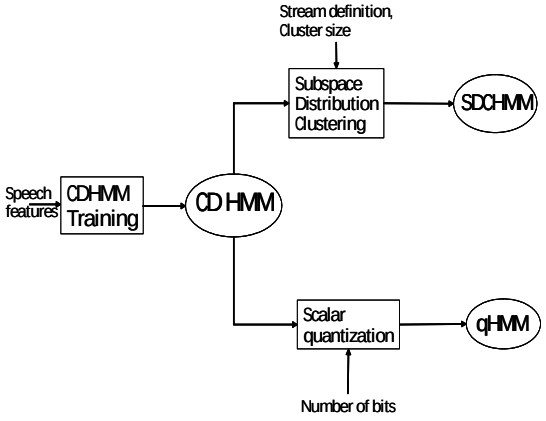


Figure 1. Derivation of qHMM and SDCHMM from CDHMM.


### 7. Acknowledgements

This work was partially been funded by the European Union under the integrated project TC-STAR - Technology and Corpora for Speech to Speech Translation -(IST-2002-FP6-506738, http://www.tc-star.org). We thank CNRS-LIMSI for providing us with the word lattices.


## 8. References


[1] Bocchieri E. and Mak B., "Subspace Distribution Clustering for Continuous Observation Density Hidden Marko Models", Proceedings of the 5th European Conference on Speech Communication Technology, Vol. 1, pp. 107-110, 1997.

[2] Bocchieri E. and Mak B., "Subspace Distribution Clustering Hidden Markov Model", *IEEE Transactions on Speech and Audio Processing*, 9(3) pp. 264-275, March 2001.

[3] Webb A., *Statistical Pattern Recognition*, Arnold, 1999.

[4] Vasilache M., "Speech Recognition Using HMMs with Quantized Parameters", Proceedings of the International Conference on Spoken Language Processing, Vol.1, pp. 441-443, Beijing, China, 2000.

[5] Vasilache M., Iso-Sipilä J. and Viikki O., "On a Practical Design of a Low Complexity Speech Recognition Engine", Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Vol. 5, pp. 113-116, Montreal, Quebec, Canada, 2004.

[6] Viikki O., Bye D. and Laurila K., "A Recursive Feature Vector Normalization Approach for Robust Speech Recognition in Noise", Proceedings of the International Conference on Acoustic, Speech and Signal Processing, Seattle, WA, USA, 1998.

[7] S. Young, "The HTK Hidden Markov Model Toolkit: Design and Philosophy", *Technical Report TR.153*, Department of Engineering, Cambridge University, UK, 1993.


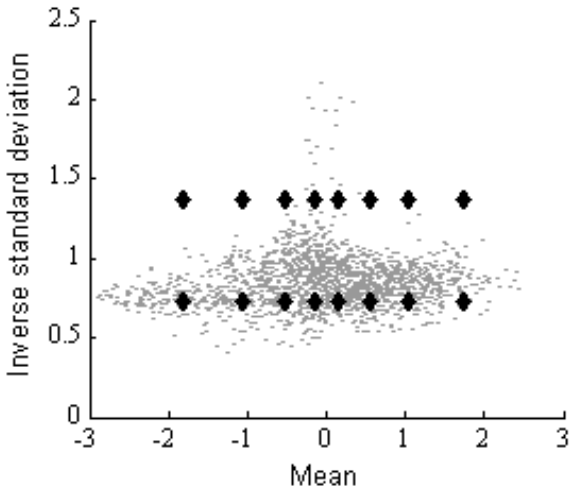


Figure 2 Scatter plot of the 2nd cepstral coefficient superimposed with the 4-bit qHMM codebook values.

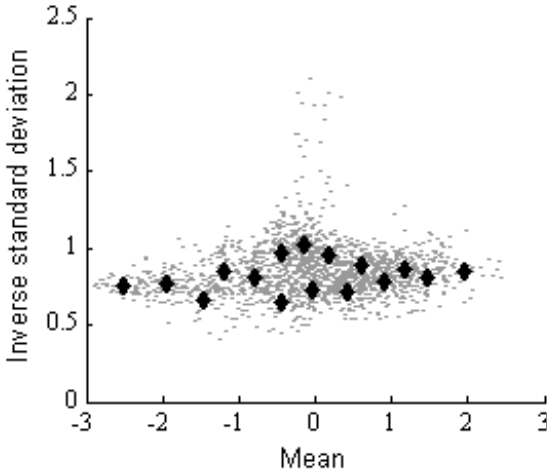


Figure 3 Scatter plot of the 2nd cepstral coefficient superimposed with the 4-bit SDCHMM codebook values.